IN THE CLAIMS

Listing of Claims:

1. (currently amended) A networking application processor, comprising:

an input socket configured to receive data packets;

a memory for storing instructions;

circuitry configured to access data structures associated with a processing stage, the circuitry configured to access data structures enabling a single cycle access of an operand from a memory location;

an arithmetic logic unit (ALU); and

circuitry for aligning operands to be processed by the ALU, the circuitry for aligning operands causing the ~~operand~~ operands to be aligned by a lowest significant bit, wherein the circuitry for aligning the ~~operand~~ operands supplies an extension to the ~~operand~~ operands to allow the ALU to process different size operands.

2. (currently amended) The networking application processor of claim 1, wherein the instructions have a width of 96 bits, and wherein the single cycle access enables the data to be addressed and operated on in a single clock cycle without being placed into a register.

3. (original) The networking application processor of claim 1, wherein the different size operands are selected from the group consisting of 8 bit operands, 16 bit operands, and 32 bit operands.

4. (original) The networking application processor of claim 1, further including:

an output socket for transmitting processed data; and

a 64 bit bus connecting the input socket and the output socket.

5. (original) The networking application processor of claim 1, wherein the extension to the operand fills each higher bit with a value.

6. (original) The networking application processor of claim 1, wherein the operand is selected from the group consisting of a source operand, a destination operand, an immediate operand, and an internal register operand.

7. (original) A processor, comprising:

an input socket configured to receive data packets;

a memory for storing instructions;

circuitry configured to access data structures associated with a processing stage, the circuitry configured to access data structures enabling a single cycle access from a memory location; and

an arithmetic logic unit (ALU), the ALU configured to receive a first and a second operand; the second operand being specified from an internal register, the first operand having a mask enabling the ALU to process a non-masked segment of the first operand.

8. (currently amended) The processor of claim 7, wherein the instructions have a width of 96 bits, and wherein the single cycle access enables the data to be addressed and operated on in a single clock cycle without being placed into a register.

9. (original) The processor of claim 7, wherein each of the instructions include a loadback feature enables random accesses to one of a source indirect register or a destination indirect register through indirect addressing.

10. (original) The processor of claim 7, wherein the mask is associated with an immediate value of the first operand.

11. (original)The processor of claim 7, wherein the first and the second operands are associated with a size selected from the group consisting of 8 bit operands, 16 bit operands, and 32 bit operands.

12. (original) The processor of claim 7, wherein the first operand is selected from the group consisting of a source operand, a destination operand, an immediate operand, and an internal register operand.

13. (original) The method of claim 7, wherein the memory location is a static random access memory (SRAM).

14. (currently amended) A processor capable of processing a data packet associated with a processing stage of a pipeline of processors, the processor comprising:

a data random access memory (RAM) configured to enable access to data structures;

instruction fetch and decode circuitry configured to interpret instructions to be executed by an arithmetic logic unit (ALU) , the instruction fetch and decode circuitry including,

a read only memory (ROM), the ROM configured to store code common to each processing stage associated with a pipeline of processors;

a code RAM, the code RAM configured to download code specific to the processing stage and wherein the code specific to the processing stage is enabled for single cycle access; and

instruction decode circuitry configured to recognize operating instructions;

execute and write back circuitry configured to set up operands to be processed by the ALU, the execute and write back circuitry including,

internal registers for defining a first and a second operand;

an arithmetic logic unit for processing the first and second operands; and

align function circuitry for aligning the first and the second operands to be processed by the ALU, the align function circuitry the circuitry causing the first and the second operands to be aligned by a lowest significant bit, wherein the align function circuitry supplies an extension to the each of the operands to allow the ALU to transparently process different size operands.

15. (original) The processor of claim 14, wherein the extension to each of the operands fills each higher bit with a value.

16. (original) The processor of claim 14, wherein the different size operands have a width selected from the group consisting of 8 bits, 16 bits, and 32 bits.

17. (original) The processor of claim 14, wherein the operating instructions wherein the operating instructions are formatted as 96 bit instructions, each of the 96 bit instructions including a single return bit.

18. (original) The processor of claim 14, wherein the processor is configured as a two stage pipeline for pipelining an instruction fetch and decode operation and an execute and write back operation.

19. (original) The processor of claim 14, wherein the operating instructions include microcode configured to predict a likely direction for a branch instruction.

20. (original) The processor of claim 19, wherein no operation (NOP's) instructions are included, the NOP's configured block an invalidated pre-fetched instruction.